Chapter

Network-on-Chip Topologies: Potentials, Technical Challenges, Recent Advances and Research Direction

Isiaka A. Alimi, Romil K. Patel, Oluyomi Aboderin, Abdelgader M. Abdalla, Ramoni A. Gbadamosi, Nelson J. Muga, Armando N. Pinto and António L. Teixeira

Abstract

Integration technology advancement has impacted the System-on-Chip (SoC) in which heterogeneous cores are supported on a single chip. Based on the huge amount of supported heterogeneous cores, efficient communication between the associated processors has to be considered at all levels of the system design to ensure global interconnection. This can be achieved through a design-friendly, flexible, scalable, and high-performance interconnection architecture. It is noteworthy that the interconnections between multiple cores on a chip present a considerable influence on the performance and communication of the chip design regarding the throughput, end-to-end delay, and packets loss ratio. Although hierarchical architectures have addressed the majority of the associated challenges of the traditional interconnection techniques, the main limiting factor is scalability. Network-on-Chip (NoC) has been presented as a scalable and well-structured alternative solution that is capable of addressing communication issues in the on-chip systems. In this context, several NoC topologies have been presented to support various routing techniques and attend to different chip architectural requirements. This book chapter reviews some of the existing NoC topologies and their associated characteristics. Also, application mapping algorithms and some key challenges of NoC are considered.

Keywords: Application mapping, interconnection, latency, scalability, system-on-chip (SoC), topology, network-on-chip (NoC), NoC design, on-chip network

1. Introduction

Distributed or parallel systems have been the main approaches of attending to the demand for applications in which huge computations are required. In these systems, a number of processing elements are connected by means of an interconnection network. It is noteworthy that the conventional off-chip architecture in which different processing elements are interconnected is unsuitable for satisfying the requirements regarding scalability, high-throughput, and low-power consumption. This is owing to the increase in delays and hardware complexities of the existing computer chips [1, 2].

The rising computational requirements of modern applications and services have shifted research attention to improvements in semiconductor-based technology. This has resulted in the evolution of on-chip networks. Based on the salient features such as low footprint and power consumption, on-chip-based networks are gaining significant attention compared to the off-chip counterpart. Instances of such on-chip-based architectures are System-on-chip (SoC) and multiprocessor-SoC (MPSoC) [3].

Furthermore, a variety of the on-chip network components are connected using various interconnection networks such as shared bus and bus. Communication among devices in bus-based topology normally occurred on the bus links, while wire collections are employed in the shared bus topology. Comparatively, the shared bus architecture offers a low-cost solution and simple control features. These benefits make the shared bus network a preferred architecture for communication among the on-chip integrated processing units [1, 4].

In addition, it is challenging for the bus-based SoC to meet the requirements of different applications due to the growing increase in the number of Intellectual Property (IP) cores as well as other on-chip resources [2]. Besides, diverse communication requirements are demanded by hybrid processing network elements. The limitations are mainly owing to the bus-based interconnection architecture's inability to offer the required latency, scalability, bandwidth, and power consumption for supporting the huge number of on-chip resources. So, the requirements are challenging for the SoC architectures to satisfy. The on-chip communication bottleneck can be effectively addressed with the implementation of network-on-chip (NoC). Apart from being able to attend to the bus architectural delays and congestion-related problems, the communication requirements can be met with lower power consumption and higher efficiency by the NoC compared to bus-based SoC [1, 4, 5].

This chapter presents a comprehensive overview of the evolution of NoC architectures and their associated features. In this regard, it focuses on a number of major and promising on-chip research areas such as topology, routing, and switching. Also, different application mapping algorithms for enhancing the on-chip performance are presented. Moreover, open problems in on-chip communication design and implementation are discussed. This chapter is organized as follows. Section 2 presents a comprehensive discussion on the SoC with a main section focus on the on-chip interconnect evolution. In Section 4, an in-depth discussion on typical NoC architectural components is presented. Besides, NoC-based application task representation and application mapping are discussed. Section 5 focuses on various NoC topology performance assessment and metrics with some models. Also, Section 6 discusses the related challenges of on-chip schemes and concluding remarks are given in Section 8.

2. System-on-chip

This section focuses on the on-chip interconnect evolution. Conceptually, SoC comprises a circuit that is embedded on a small coin-sized chip and integrated with a microprocessor or microcontroller. Moreover, in the SoC, a single chip is partitioned into functional tiles and an interconnection (communication) network. Depending on the application, the SoC design typically contains storage devices, RAM/ROM memory blocks, central processing unit (CPU), input/output ports, and peripheral interfaces such as timers, inter-integrated circuit (I^2C), universal asynchronous receiver/transmitter (UART), graphics processing unit (GPU), controller area network (CAN), serial peripheral interface (SPI), and so on. Also, based on the requirement, a floating-point unit or analog or digital signal processing system can as well be included.

Furthermore, with innovative integration technology, a huge amount of heterogeneous cores can be supported on a single chip [6, 7]. So, in designing such a chip, the interconnections between multiple cores have a considerable influence on the system performance and communication. For instance, apart from being in charge of all memory transactions, on-chip communication architecture is also responsible for I/O traffic management. Besides, it offers a reliable channel for data sharing between processors. Therefore, high-performance and scalable on-chip communications are highly imperative in the SoC design [8]. However, efficient communication among the on-chip components is challenging [1, 2, 4, 8]. **Figure 1** illustrates an on-chip interconnect architectural evolution.

2.1 Bus architecture

As aforementioned, the traditional on-chip bus-based interconnect techniques are widely used partly due to protocol and architectural design simplicities. Other salient features of bus topology are low area overhead and predictable latency. For instance, bus-based interconnect is mainly suitable in a scenario with a small amount of on-chip components. Besides, it is not only efficient regarding power but also based on the silicon cost [1, 4, 8].

Furthermore, as illustrated in **Figure 1(a)**, a single control/data bus is used in this architecture to support multiple component interactions. This helps in ensuring a simple master–slave connection. Moreover, resource contention can also occur in bus-based architecture when multiple masters are required to communicate with the same slave. In such a situation, arbitration is demanded to ensure effective communication. Therefore, in a large SoC, the bus-based architecture scalability is challenging [8].

2.2 Crossbar (matrix switch fabric) architecture

As discussed in subSection 2.1, when multiple master–slave data transactions are to be supported, a single shared bus architecture is not a suitable solution due to the associated latency. This is as a result of the arbitration that is demanded between the master interfaces on the shared medium. Consequently, crossbar topology can be employed to address the scalability issue of on-chip interconnect. As illustrated in **Figure 1(b)**, a crossbar architecture consists of a matrix switch fabric. To facilitate multiple communications, this matrix connects the entire inputs with the entire outputs. Based on the advantages of the crossbar topology, it has been adopted by the SoC designer by merging multiple shared busses to achieve a connection matrix

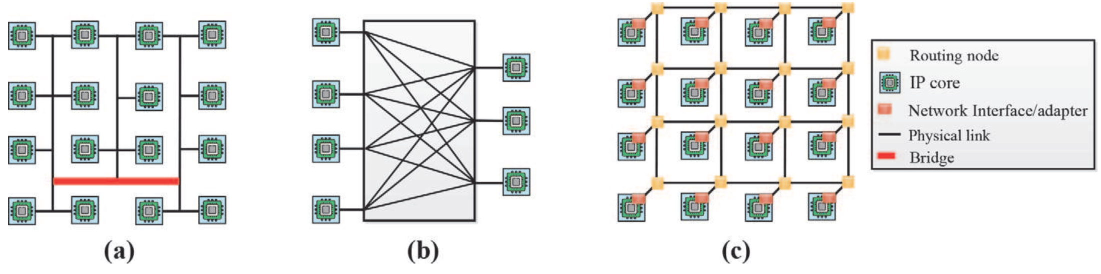


Figure 1.
On-chip interconnect architectural evolution: (a) bus, (b) crossbar, and (c) 4 × 4 mesh-based NoC.

with an all-input-all-output [8, 9]. Nonetheless, the design complexity of a crossbar-based architecture is high owing to the layout of wire that is complex [8].

2.3 Network-on-chip architecture

NoC offers an alternative and modular nature platform with high scalability. Besides, it supports efficient on-chip communication that facilitates NoC-based multiprocessor architectures with high functional diversity and structural complexity [3]. This makes it a *de facto* on-chip communication standard for highly integrated SoC architectures. Besides, it supports parallel (multiple concurrent) communications by enabling pipelining irrespective of the network size. Also, rather than establishing a connection between all IP blocks, in the NoC, a network is created within the chip. This enables each IP to function as a network node. For instance, to ensure effective communication, a network of routers is employed to connect the associated huge cores. In this regard, the bus in SoC has been replaced by a network of routers that controls the communication process among nodes in the established network. Based on this, NoC presents a number of characteristics such as low-latency, high-bandwidth, and scalability [10, 11].

In the NoC, the interconnections are suitably organized to form appropriate topologies. Moreover, communication in it normally transpires between IP cores and in accordance with the employed topology. Also, this can be achieved using asynchronous or synchronous modes [11]. So, with these topologies, certain routing techniques can be employed for packet routing between nodes [10]. As depicted in **Figure 1(c)**, components such as routers and channels (interconnection links) are required for packet routing [11]. It is noteworthy that some routing techniques are specially intended for NoC. As a result, they are well-designed to be deadlock[1]-free [10].

Furthermore, **Figure 1(c)** illustrates a 4 × 4 mesh-based NoC architecture with a number of processing cores/processing elements (PEs) connected via regular-sized wires and routers. The PEs can be components such as application-specific integrated circuit (ASIC) block and microprocessor [12]. Moreover, different types of cores such as the manager, regular, and spare can be employed. Also, depending on the application, these cores can be homogeneous and heterogeneous. The regular cores normally execute the task of a specified application, the spare cores are additional cores that can be employed in case of failure of either regular or manager core, and the manager cores are used to track and manage all processing cores. Besides, when a processing core fails, the manager core performs the task migration [13].

Moreover, a network interface (NI) is placed at the edge of each PE and on-chip interfaces such as high-definition multimedia interface (HDMI), I^2C, USB, and UART are supported. The routers are employed to packetize the generated data by the PE. The NI is subsequently connected to a router that buffers the packets from the PE or other connected routers [12]. In this regard, the NI connects the NoC routers and hardware IP blocks. Consequently, NI facilitates the modular property and ensures seamless communication between different IP with related housekeeping operations, irrespective of their communication protocol [8].

In addition, it is noteworthy that the experienced design problems in the NoC architectures are similar to the ones in the bus-based architectures. In this context, to establish a communication fabric capable of meeting the requirements of a particular application, a trade-off between reliability, power, area, cost, and performance is demanded [12]. For instance, asymptotic cost functions for a shared

[1] A deadlock happens in the NoCs when one or more packets remain blocked for an indefinite time. It can be addressed either by imposing routing restrictions or by employing additional hardware resources.

	Power Dissipation	Operation Frequency	Total area
Shared bus	$\mathcal{O}(n\sqrt{n})$	$\mathcal{O}(\frac{1}{n^2})$	$\mathcal{O}(n^3\sqrt{n})$
Segmented bus	$\mathcal{O}(n\sqrt{n})$	$\mathcal{O}(\frac{1}{n})$	$\mathcal{O}(n^2\sqrt{n})$
Point-to-Point	$\mathcal{O}(n\sqrt{n})$	$\mathcal{O}(\frac{1}{n})$	$\mathcal{O}(n^2\sqrt{n})$
NoC (Mesh)	$\mathcal{O}(n)$	$\mathcal{O}(1)$	$\mathcal{O}(n)$

Table 1.
Asymptotic cost functions for interconnection architectures.

bus, segmented bus, point-to-point, and NoC interconnect with n system modules are presented in **Table 1**. This shows that with a growing n, NoC architecture dissipates less power, requires less wiring area, and offers excellent operating frequency, making it a scalable and attractive architecture [14].

3. Advanced on-chip bus architectures

Since there are various IPs with distinct standard interfaces from different providers, the chip designers have to adapt to connect through common standard or in-house interfaces. Consequently, a flexible and open standard for IP core interface is essential for practical on-chip interconnection design and SoC integration. This can be achieved by employing standard interface protocols that offer reusable profiles that can support diverse on-chip interconnection design and SoC integration. Also, the operation of an on-chip interconnection depends on the bus architecture efficiency. So, bus architecture with additional data transfer cycle, faster clock speed, enhanced throughput, and width is highly attractive for a reduced time-to-market, low cost, and efficient SoC. This section presents an overview of standard on-chip bus structures and protocols such as ARM Advanced Microcontroller Bus Architecture (AMBA), IBM CoreConnect, and Altera Avalon.

3.1 AMBA-based bus protocol architecture

The AMBA bus protocols are the ARM interconnect specifications for on-chip communication between a number of functional blocks. In these designs, one or more microprocessors/microcontrollers can be integrated on a single chip with various other components and peripherals. **Figure 2** depicts a traditional AMBA 2.0 based SoC design with Advanced System Bus (ASB) or Advanced High performance (AHB) protocols and an Advanced Peripheral Bus (APB) protocol for high bandwidth and low bandwidth peripheral interconnections, respectively [15].

Moreover, to scaling up connectivity and address the limitations regarding the number of IPs that can be effectively supported by the AHB/ASB protocols, AMBA 3 presents Advanced Extensible Interface (AXI) interconnect for point-to-point

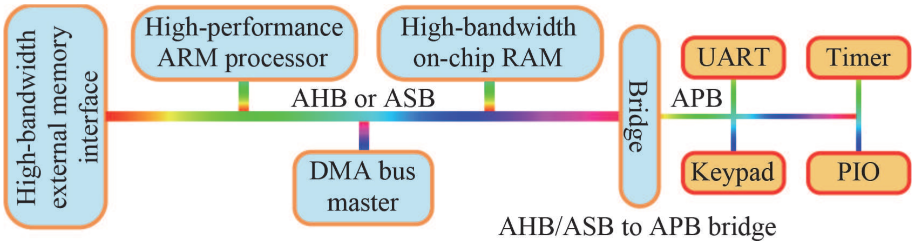

High-bandwidth external memory interface
High-performance ARM processor
High-bandwidth on-chip RAM
AHB or ASB
DMA bus master
Bridge
APB
UART
Timer
Keypad
PIO
AHB/ASB to APB bridge

Figure 2.
A typical AMBA based SoC design. PIO: Peripheral I/O, UART: Universal asynchronous receiver/transmitter,

connectivity protocol. Some of the main features of the AXI protocol are the capability to issue multiple outstanding transactions, unaligned data transfers with byte strobes, separate control/address and data phases, simultaneous read and write data channels to guarantee low-cost Direct Memory Access (DMA), and out-of-order data capability. **Figure 3** shows AXI interconnect enabling IPs communication with a master–slave protocol. It is noteworthy that the interconnect such as a switch design, a convention crossbar, or an off-the-shelve NoC capable of supporting multiple AXI masters and slaves can be employed. Also, an array of peripherals supported on an APB bus are connected through an AXI to APB bridge [15, 16].

Furthermore, the emergent of portable mobile devices such as smartphones in which SoCs are equipped with dual/quad/octa-core processors and shared integrated caches demand hardware managed coherency within the memory subsystem, resulting in the development of AXI Coherency Protocol Extension (ACE) in the AMBA 4. Also, with the current trend towards heterogeneous computing for improving the performance of data center, parallel, and High-Performance Computing (HPC) applications, numerous heterogeneous computing elements, processor cores, and IO subsystems are demanded. To support the requirements, the Coherent Hub Interconnect (CHI) protocol was presented in the AMBA 5 protocol to improve the AXI/ACE protocol design. For instance, for better scalability, the associated signal-based protocol in the AXI/ACE was changed to a packet-based layered protocol in the CHI. Some of the supported features are Cache stashing, Cache de-allocation transactions, atomic transactions, and Persistent Cache Maintenance Operations (CMO). Other AMBA specifications with additional efficient translation services and higher performance are Distributed Translation Interface (DTI) and Local Translation Interface (LTI) protocols [16, 17].

3.2 WishBone bus protocol architecture

WishBone interconnect primarily focuses on design reuse to address integration problems by establishing a general-purpose interface between IP cores. This helps in improving the system's portability and reliability. This interconnect comprises two interfaces which are master and slave. The IPs are master interfaces that can initiate bus cycles. Also, the slave interfaces accept the initiated bus cycles. Besides, its hardware implementations are compatible with varieties of interconnection such as dataflow, crossbar-switch, shared bus, and point-to-point [16].

Furthermore, WISHBONE specifies a single, simple, logical, synchronous MASTER/SLAVE bus and IP core interfaces that demand very few logic gates. Also, it supports some standard data transfer protocols such as BLOCK READ/WRITE cycles, SINGLE READ/WRITE cycles, and read-modify-write (RMW) cycles.

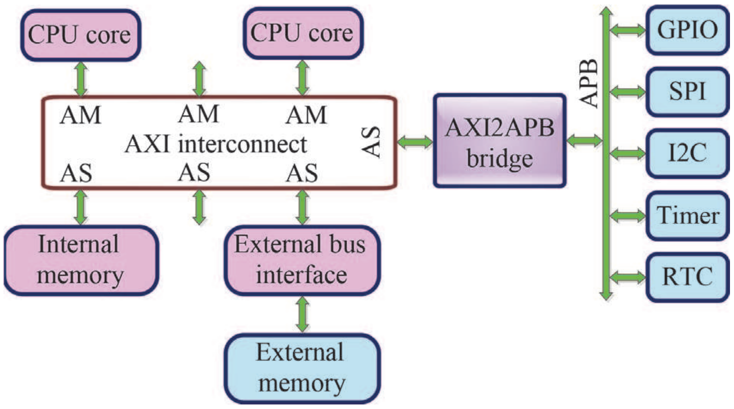


Figure 3.
An AXI interconnect. GPIO: General purpose input/output.

Moreover, the related flow control and communication among the cores are facilitated by the handshake mechanism. Besides, its multiprocessing capabilities enable a broad range of SoC configurations [17, 18].

3.3 Open core protocol

Open core protocol (OCP) is an open standard, non-proprietary, core-centric protocol for attending to the requirements of IP core system-level integration. Also, it defines a clocked system that offers unidirectional data transfer that helps in simplified core integration, implementation, and timing analysis. Moreover, based on its high configurability and flexibility, it supports independent IP cores design and facilitates IP reuse. Based on this, it enhances and ensures IP modularity without the need for redesign. Besides, all test/debug and sideband signals are offered by the OCP for a number of functions, such as protections or interrupts. Also, some of its features and signals are optional. This helps the users in choosing the configuration that best suits their IP cores [17].

A typical OCP operation across an on-chip interconnect is shown in **Figure 4**. In this configuration, an OCP master/slave element is integrated into IP cores. The implementation comprises a request and a response channel. The master IP core issues read command that causes a transfer on the request channel. On the response channel, the slave IP core responds to the master IP core. Also, some supported extensions by the OCP protocol are the transfer of a burst of data, data handshake extensions, out-of-order responses, and test control extension [16].

3.4 IBM CoreConnect architecture

IBM CoreConnect™ architecture is another open on-chip bus structure that offers the framework for efficient realization of complex SoC designs. As illustrated in **Figure 5**, it has three separate busses for interconnecting cores, custom logic, and

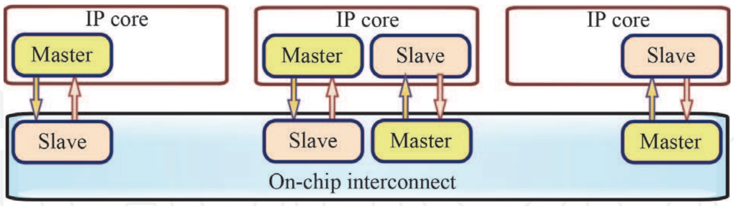


Figure 4.
Typical block diagram of OCP.

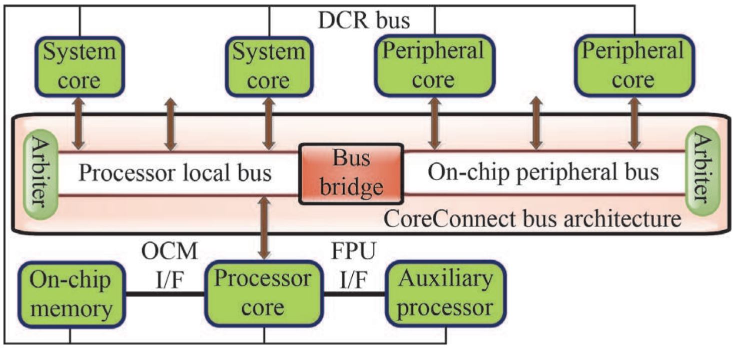


Figure 5.
Typical block diagram of CoreConnect.

library macros are On-Chip Peripheral Bus (OPB), Processor Local Bus (PLB), and Device Control Register Bus (DCR). The architecture can be employed for different customer-specific and application-specific SoC designs in high-performance embedded applications, storage, networking, wired/wireless communications, and low-power pervasive applications. In this context, high-performance peripherals can be connected to the low latency, high bandwidth PLB. Also, device-paced peripheral cores are normally connected to the OPB. This helps in reducing traffic on the PLB and consequently, enhancing the system performance. Also, a relatively low-speed data path is offered by the DCR bus for control, initialization, and status information [16, 18].

3.5 Altera Avalon architecture

Avalon presents a simple bus architecture for the connection of on-chip peripherals and processors to a system-on-a-programmable chip (SOPC). Also, the offered interface defines a port for connecting the master and slave components and the timing for the components' communication. Besides, it supports multiple masters that present construction flexibility for SOPC systems. The slave-side arbitration is used in the masters and slaves interaction. So, if multiple masters try to access the same slave simultaneously, slave-side arbitration logic controls the master that gains access to the slave to complete the requested transactions. **Figure 6** illustrates a typical block diagram of an Avalon bus module with a collection of connected peripherals [17, 19].

Moreover, the Avalon bus module comprises data, address, and control signals, and arbitration logic that are needed for connecting the peripheral components. Also, its operation comprises address decoding for the selection of peripheral and wait-state generation for supporting slow peripherals. Furthermore, apart from the simplicity and optimized resource utilization, the bus also offers synchronous operation and dynamic sizing. Also, different transactions can be realized between a master and slave peripheral. Likewise, different advanced features such as multiple bus masters, streaming peripherals, and latency-aware peripherals are supported. Based on this, during a single bus transaction, multiple data units can be conveyed between peripherals [18, 19]. **Table 2** compares different SoC busses.

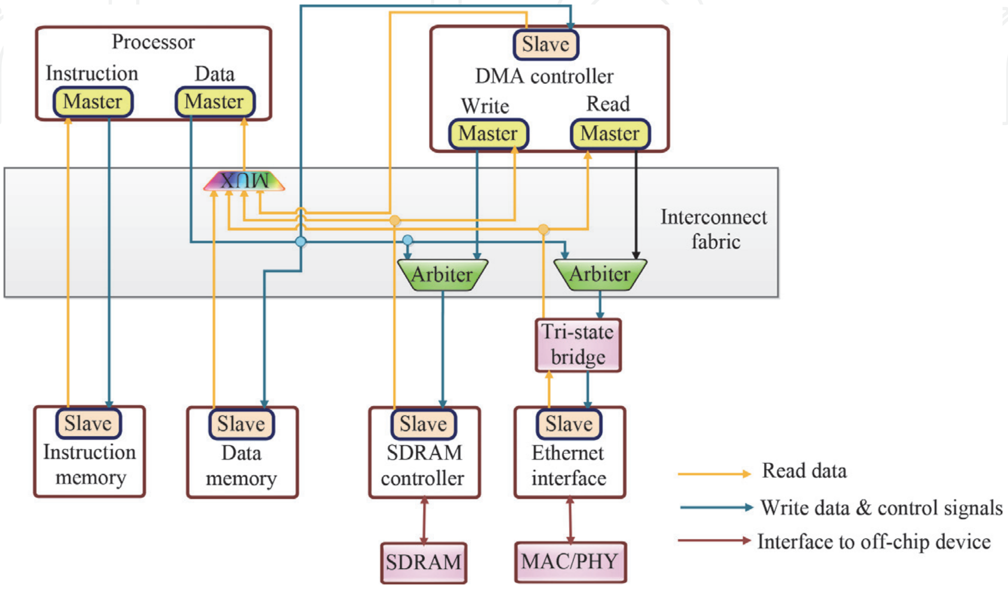


Figure 6.
Typical block diagram of Avalon.

Protocol	Bus Owner	Bus Topology	Arbitration	Bus Width (bits)		Transfers
				Data	Address	
AXI	ARM	Bus-Matrix & Hierarchical	Static Priority, TDMA, Lottery, Round-Robin, Token-passing and CDMA	8, 16, 32, 64, 128, 256, 512, or 1024	32	Handshaking, Split, Pipelined and Burst
Wishbone	OpenCores.org & Silicore Cooperation	Point-to-Point, Crossbar Connection, Shared & Data-flow Interconnection	Static Priority, TDMA, Lottery, Round-Robin, Token-passing and CDMA.	8,16,32,64	1–64	Handshaking & Burst
OCP	OCP Int. Partnership	Interconnect Topology	Vary depending on logic on the bus side of OCP.	Configurable	Configurable	Handshaking, Split, Pipelined & Burst
Avalon	Altera	Point-to-Point, Pipelined, Multiplexed	Static Priority, TDMA, CDMA, Round-Robin, Lottery, Token-passing	1–128	1–32	Pipelined and Burst
CoreConnect	IBM	Hierarchical	Static Priority	PLB (32, 64, 128 or 256); OCB (8, 16 or 32) and DCR (32)	PLB and OPB (32) and DCR (10)	Handshaking, Split, Pipelined and Burst

Table 2.
Comparison of SoC busses.

4. Network-on-chip components

As aforementioned, a network of routers is employed in the NoC for controlling the communication process among nodes. Several topologies along with different routing algorithms have been presented for NoC architectures. It is noteworthy that the network topology selection is a primary step in the network design. Besides, the flow-control techniques and routing strategy depend greatly on the topology. This section focuses on typical architectural components such as network topology, switching, and routing algorithm. Besides, task representation and application mapping are presented.

4.1 Network-on-chip topologies

The NoC topology denotes the physical organization of its architecture and it signifies a key design criterion. In this context, the means by which its elements are interconnected are characterized. The NoCs have been considered as regular tile-based topologies that are appropriate for connecting homogeneous cores. Besides, much attention has been given to custom-based, domain-specific irregular topologies to support heterogeneous cores with diverse size, functionality, and communication requirements [3]. Some of these topologies are discussed in this subsection.

4.1.1 Regular topologies

In regular topologies, the power consumption and network area scalability with an increase in the size can be predicted. It should be noted that regular network topologies are usually adapted for the majority NoCs [20]. This subsection focuses on the most popular regular topologies along with their advantages and drawbacks.

Ring Toplogy:

Ring topology is one of the widely employed NoC topologies. In this topology, a single wire is used to connect each node. Consequently, irrespective of the ring size, each of the nodes has neighboring nodes as depicted in **Figure 7(a)**. Based on this, in the ring topology, the degree[2] of each node is two. This implies a corresponding available bandwidth to every node. Although deployment and troubleshooting are comparatively easier, the main drawback of the ring topology is that its diameter increases with an increase in the number of nodes. So, besides the fact that network expansion degrades the performance (scalability issue), ring topology is also prone to a single point of failure (poor path diversity) [1, 21].

Octagon Topology: Another prevalent NoC topology is the octagon. A typical octagon topology comprises eight (8) nodes and twelve (12) bidirectional links. Also, just like the ring topology, each node is connected to the preceding and succeeding nodes. So, between a node pair, there are two-hop communications. Also, to route a packet between the network, a simple shortest-path routing can be employed. Besides, compared with a shared bus topology, higher aggregate throughput can be achieved. Furthermore, the architecture can be connected to support bigger designs, resulting in better scalability.

Star Topology: The star topology in which the entire nodes are connected to a central node is shown in **Figure 7(c)**. Assume an N nodes with $N - 1$ connected nodes to the central node. In this architecture, the central node has an $N - 1$ degree

[2] Router degree is a parameter that specifies the number of on-chip components and neighboring routers that it is connected to. It is noteworthy that the router microarchitecture complexity increases with an increase in its degree.

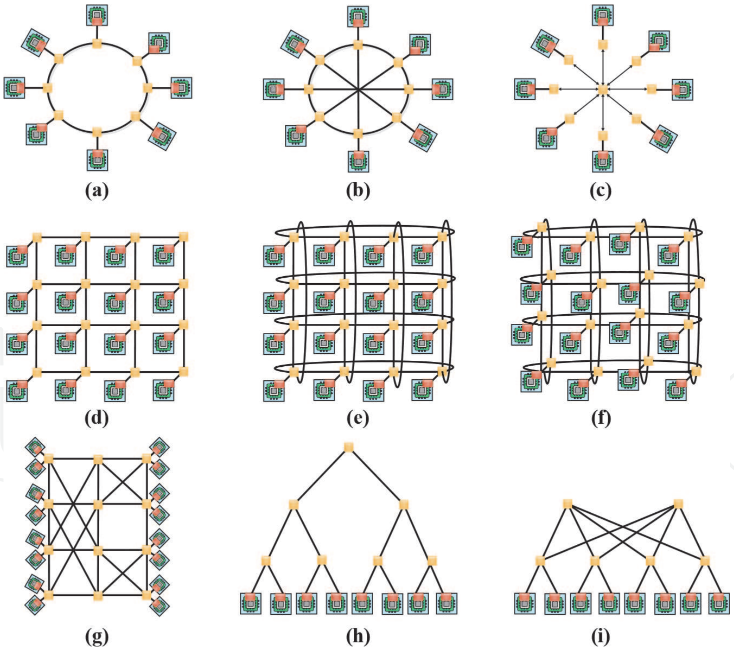


Figure 7.
NoC topologies: (a) ring, (b) octagon (c) star, (d) 4 × 4 mesh, (e) 4 × 4 torus, (f) 4 × 4 folded torus, (g) butterfly, (h) binary tree, (i) fat tree.

while others have a degree of 1. Therefore, regardless of its size, the star topology diameter is 2. In this regard, its main benefit is the offered simplicity and the presented minimum hop count of two due to the associated small diameter [21]. Although the nodes are separated and free of the potential impact from the failed nodes, the central node failure can result in the entire network failure. Furthermore, as the diameter of the central node increases with the number of nodes, a communication bottleneck can take place in the central node [1].

Mesh Topology: The mesh architecture is the widely employed interconnection topology. A typical 4 × 4 mesh topology with 16 functional IP blocks is illustrated in **Figure 7(d)**. Besides the router at the edges, each router in the mesh topology is connected to one computation resource and four neighboring routers through communication channels. With mesh topology, a huge number of IP cores can be incorporated in a regular-shape structure [4]. So, this topology offers an attractive solution for path diversity and scalability. Likewise, this topology can tolerate link failure due to multiple paths that connect a pair of nodes [21]. Nevertheless, one of the main challenges of this topology is that its diameter increases significantly with the number of nodes. This is owing to irregularity in the degree [1]. For instance, the degree of corner, edge, and inner nodes are 2, 3, and 4, respectively [21]. Besides, the associated bandwidth often varies from one node to another, with corner and edge nodes having lesser bandwidth [1, 21].

Torus Topology: A typical torus topology is depicted in **Figure 7(e)**. The architecture is very similar to a mesh topology. However, mesh topology offers a reduced diameter. Consequently, the challenge of diameter increase of mesh topology with the network size is addressed by the torus topology. This is achieved through the addition of direct connections between the end nodes that are in the same column or row [21]. For instance, in the torus topology, wrap-around channels are employed for the connection of the edge routers to those at the opposite edge, resulting in a better bisection bandwidth and reduced average number of hops. However, considerable latency is incurred by the torus topology due to the employed lengthy wrap-around connections [1, 4].

In addition, an alternative to the torus is the folded torus topology. The folded torus topology offers a shorter link length, resulting in reduced implementation area and traverse time for the packet between the interconnected links. Compared with the torus, folded torus offers more path diversity, making it more fault-tolerant. Besides, as aforementioned, torus topology helps in reducing the associated mesh latency. Nevertheless, its long wrap-around links can cause undue delay. This challenge can be addressed by folding the torus as depicted in **Figure 7(f)**.

Butterfly Topology: A typical butterfly topology is illustrated in **Figure 7(g)**. It offers a fixed hop distance between any source to the destination node pair and the router degree is 2, resulting in low-cost routers. Owing to the single path that exists from the source to the destination node, the topology lacks path diversity, resulting in low link fault tolerance and low bandwidth. Besides, this topology normally entails lengthy wires and the related complex wire layout can lead to more energy consumption [1, 21].

Binary Tree Topology: The binary tree topology consists of a top (root) node and bottom (leaves) nodes illustrated in **Figure 7(h)**. In this configuration, besides the root node, each of the others has two offspring nodes. Also, besides the root node that has no parent, each of the other nodes has its parent and children directly above and beneath itself, respectively. The nodes in this topology have access to broader network resources and it is supported by several vendors. However, its bottleneck is the root node, whose failure can bring about the entire network failure. Also, with an increase in the tree length, network configuration becomes more intricate.

Fat Tree Topology: The concept of fat tree topology is based on using intermediate routers as forwarding routers and connecting the leave routers to the clients as illustrated in **Figure 7(i)**. Although this topology offers excellent path diversity and better bandwidth, the router to clients ratio is extremely high and the wiring layout is complex. Therefore, a number of routers should be integrated to connect with fewer clients [1].

Cube-Based Topology: There are a number of cube-based topologies that have been designed. One such appropriate architecture is a hypercube topology. However, its main disadvantage is that there are restrictions in its network size because of the degree limitation. To address the limitation, various variations such as folded hypercube, dual cube, crossed cube, cube-connected cycles, hierarchical cube, and metacube have been presented. A number of these topologies are depicted in **Figure 8** and are mainly focusing on reducing the associated node degree and/or minimizing the network diameter while the diameter is kept as small as possible [10, 21, 22].

In a folded hypercube, each node is connected to the farthest distinct node. Based on this, there is a considerable reduction in its diameter compared with the hypercube topology. However, this is at the expense of additional links. Furthermore, a crossed cube can be realized through the transposition of some edges in the hypercube. This helps in the diameter reduction without causing additional link complexity. In a reduced-hypercube, to minimize node degree, the edges are reduced from an n-dimensional hypercube [22]. An (n, n) hierarchical cubic network consists of n cluster and each of the clusters has n-cube. Furthermore, a hierarchical hypercube is a dual cube structure. This topology comprises two classes

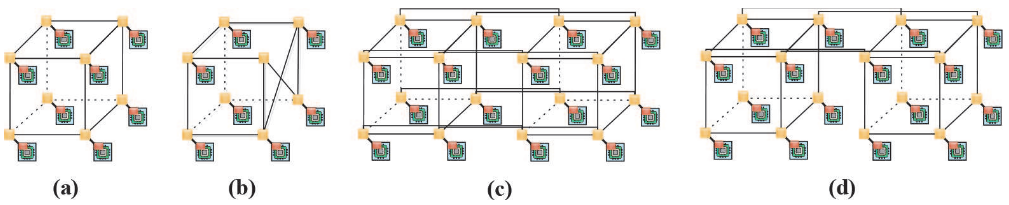


Figure 8.
Cube-based topologies: (a) cube, (b) crossed cube, (c) hypercube and (d) reduced hypercube.

(0 and 1) with clusters. Also, each of the clusters contains $2m$ nodes. Likewise, in an m-dual cube, the binary address of each node is $1 + 2m$ bit long. Similarly, cube-connected cycles offer a hypercube implementation with virtual nodes. In this topology, rather than a single node, each virtual node is a circle with three ports. Also, a metacube topology is a two-level hypercube architecture. It is a symmetric network with a short diameter and small node degree. Structurally, this multi-class topology is an extended form of the dual cube [21].

4.1.2 Irregular topologies

Irregular topologies are based on the integration of various forms, usually regular structures, in different fashions. In this regard, a hybrid, hierarchical, or asymmetric approach can be adopted. Moreover, irregular topologies aims at increasing the available bandwidth compared with the traditional shared busses. Besides, compared with the regular topologies, it helps in reducing the distance among nodes [12]. Also, irregular topologies typically scale nonlinearly with area and power. They are usually based on the concept of clustering and adapted for specific applications [20]. **Figure 9** illustrates some irregular topologies such as reduced (optimized) mesh (**Figure 9(a)**- i and ii) and cluster-based hybrid (mesh + ring-**Figure 9(b)**).

In addition, apart from the classification discussed in subSection 4.1.2, NoC topology can also be categorized as direct[3] and indirect[4] topologies. For instance, ring, bus, mesh, and torus topologies are direct topology. On the other hand, a clos, butterfly, benes, and fat-tree are good instances of indirect topology [11, 12].

4.2 Network-on-chip routing

With suitable topology, a network will be established among the on-chip IPs to ensure effective communication. The communication can be achieved using appropriate algorithms for routing the packets from the source to the destination nodes. In this context, routing algorithms control efficient and correct packet routing as they traverse through the nodes. As aforementioned, starvation[5]-free and deadlock-free routing algorithms are of utmost importance in NoC [10]. Furthermore, the routing algorithm can be selected based on a number of interrelated features,

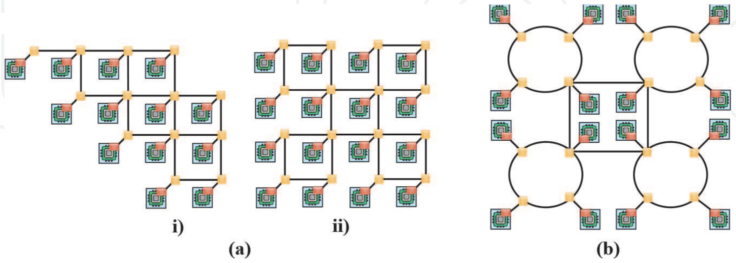


Figure 9.
Irregular topologies: (a) reduced mesh structures and (b) cluster-based hybrid.

[3] In this topology, there is a direct connection of each router to at least a core.

[4] In this topology, some of the employed routers are not directly connected to any of the cores.

[5] Starvation usually occurs in NoCs when specified priority rules are employed for routing, mainly in favor of the high priority packets, making low priority packets wandering in the network. It can be attended to by reserving some resources for the low priority packets and adopting fair routing algorithms.

resulting in trade-offs between related metrics such as the power consumption, packet latency, and footprint that determine the routing algorithm quality. For instance, when the routing circuit is kept simple, the required power for routing can be minimized, and consequently, the power consumption can be reduced. Besides, to increase the performance, the routing tables should be minimized. This will help in ensuring low latency, enhanced robustness low footprint, and effective network utilization [8, 12]. In general, the NoC routing algorithms can be classified based on factors such as the routing path, distance, and decision states. In this context, its NoC routing algorithms can be mainly classified as static and dynamic routing algorithms. Besides, the routing decisions can also be based on distributed, source, minimal, and non-minimal routing algorithms. This subsection focuses on the static and dynamic routing algorithms.

4.2.1 Static routing

Static routing, also known as the oblivious or deterministic routing is the simplest and extensively used routing algorithm in NoCs. It employs fixed (predefined) paths for data transfer between a specific source and destination. Also, the current state of the network is not taken into account in the static routing. So, when making routing decisions, it is oblivious of the load on the links and routers. Static routing requires very little router logic, making its implementation easy. Besides, packets can be split in a scheduled way among several paths between the source and destination. Also, in-order packet delivery can be guaranteed by the static routing in a scenario where just a path is employed. Based on this, the addition of bits to packets at the NI is not required for correct identification and reordering at the destination [12]. Schemes such as random walk routing, directed flood, probabilistic flood, dimension order routing (DOR), destination tag, turn model, XY, surrounding XY, and pseudo-adaptive XY, are examples of static routing algorithms.

XY routing is a distributive deterministic routing algorithm. In this algorithm, the coordinates of the destination address are employed in delivering the packet through a network. The packet is initially routed along the X coordinate (horizontal direction) to reach the column. Then, is routed vertically along the Y coordinate to its destination [5]. XY routing is a preferred algorithm for torus- based and mesh-based topologies [10, 11] and it is deadlock-free. Nevertheless, the associated traffic can be irregular due to the load that is normally created in the middle of the network [10] while XY algorithm is not capable of avoiding congested and busy links [5].

4.2.2 Dynamic routing

The routing decisions in adaptive or dynamic routing are based on the existing state of the underlying network. To make routing decisions in this routing scheme, factors like system availability and load condition of the links are considered. Consequently, as the application requirements and traffic conditions change, there can be a corresponding change in the path between the source and destination. Compare with static routing, traffic can be distributed more efficiently across various routers in dynamic routing. Besides, in case of network congestion in certain NoC links, it can exploit alternative paths. In this regard, more traffic can be supported by its topology and the network bandwidth utilization can be maximized. These salient features of the adaptive routing are owing to its ability to exploit the global knowledge of the current traffic state in the optimal path selection [23]. However, this scheme's adaptivity is at the expense of additional resources required for continuous monitoring of the network state to ensure a corresponding dynamic change in the routing paths. This usually presents additional complexity to the router design. Besides, there is a limitation on the effectiveness of adaptive routing due to the constraint on the amount of global knowledge that can be forwarded to each of the routers and also owing to interference [23]. As aforementioned, a static routing scheme is normally employed in scenarios with steady and known traffic requirements, while dynamic routing is primarily applicable to unpredictable and irregular traffic conditions [8, 12]. Schemes such as congestion look-ahead, slack time aware, fully adaptive, minimal adaptive, turnback when possible, turnaround–turnback, odd-even, deflection (hot potato), are examples of adaptive routing algorithms.

In the NoC, to communicate between the source and destination, both adaptive and deterministic routing algorithms can be employed. The odd-even routing algorithm is an adaptive routing and it is a deadlock-free turn model. In this regard, in a grid network, east to south and east to north turns are prohibited in the even columns. Also, north to west and south to west turns are prohibited in the odd columns. So, the odd-even routing algorithm helps in eliminating potential livelock[6] in the system [5, 10].

The deflection routing technique is cost-effective since no buffers are employed. Consequently, the incoming packets received by the routers are not buffered and move simultaneously towards their destinations based on the routing table. However, misrouting can occur when a busy router receives another packet. In a severe situation, the misrouted packets in the network can cause additional misrouting, making each packet to be bouncing around like a *hot potato* across the network [12]. The misrouting can be alleviated considerably with sufficient intervals between the packets [10].

4.3 Network-on-chip switching

The NoC switching scheme denotes the employed switching technique for data control in the routers and specifies the data transfer granularity. Packet switching and circuit switching are the key switching techniques in the NoCs. The switching schemes are illustrated in **Figure 10** and discussed in this subsection.

4.3.1 Circuit switching

The circuit switching is based on the establishment of a reserved physical path (link reservation), consisting of routers and links, between the source and destination before data transmission. Although circuit switching offers low latency transfers due to the full link bandwidth utilization, it wastes the established links when there is no

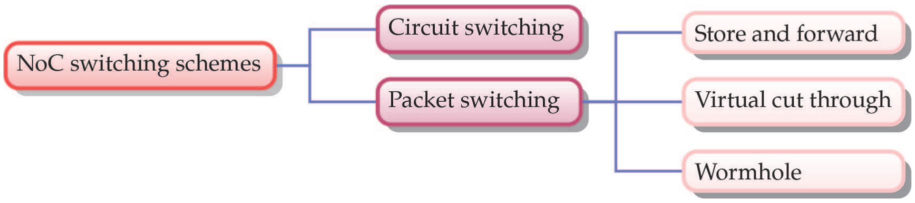


Figure 10.
NoC switching schemes.

[6] A livelock arises in NoCs when a packet bounces around indefinitely between routers without reaching its destination. It is typically associated with adaptive routing and can be addressed by employing uncomplicated priority rules.

data transmission, resulting in scalability issues. Furthermore, to improve network scalability, virtual-circuit switching can be employed. It helps in multiplexing multiple virtual links on a single physical link. Also, the allocated buffers determine the total number of virtual links that the physical link can support [12].

4.3.2 Packet switching

Packet switching is another popular switching mode. Unlike circuit switching in which a path is established prior to the data transmission, in packet switching, there is no need to create a path (no link reservation) before packet transmission. In this context, the transmitted packets follow independent paths (different routes) from the source to the destination. As a result, different delays will be experienced by the packets. Besides, unlike circuit switching in which a start-up waiting time and a fixed minimal latency are normally incurred, in packet switching, a zero start-up time and a variable delay owing to contention are generally incurred. Moreover, due to the contention, Quality of Service[7] (QoS) in packet switching is challenging to guarantee compared with circuit-based switching. Wormhole, virtual cut through as well as store and forward are the extensively employed packet switching schemes [12].

4.4 Network-on-chip application mapping

In supercomputing and parallel computing, application mapping is usually employed for mapping applications that share resources to be in close proximity to minimize the network latency. This is also applicable to the shared bus-based Chip Multi-Processor architectures in which the application mapping should consider the fundamental on-chip interconnect design. Depending on the adopted topology, mapping algorithm implementation helps in the positioning of the IP cores to the NoC tiles. Besides, its performance is highly contingent on the employed routing interface and shared memory architecture [8].

In MPSoC, quite a lot of techniques can be employed for application mapping. Also, the presence of several and diverse MPSoC architectures further complicate the issue. Consequently, in practice, it is advisable to rebuild the mapping approach based on the application-architecture category [8]. Furthermore, as depicted in **Figure 11**, the NoC application mapping algorithms are broadly grouped into static and dynamic based on the assigned task time. In this context, the time at which the tasks are allocated to the IP cores for processing is considered. For instance, in dynamic mapping, the application task clustering, ordering, and assignment to the cores are

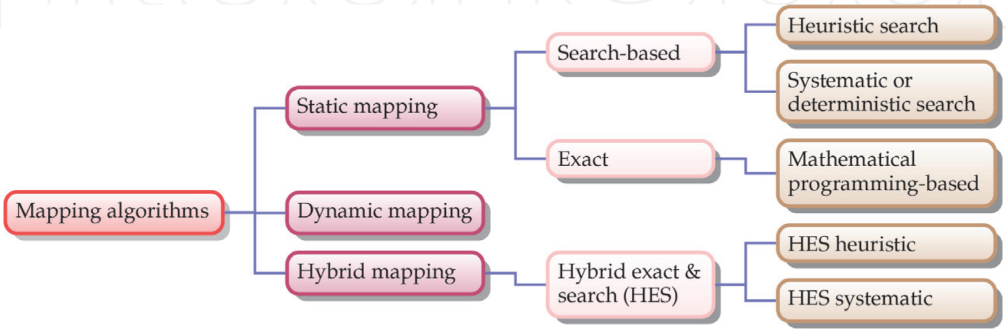


Figure 11.
NoC mapping algorithms classification.

[7] Quality of Service implies performance bounds regarding the delay, bandwidth, and jitter; and can be categorized into differentiated service, guaranteed service, and best effort.

implemented in the course of application execution (real-time). Also, dynamic mapping is an efficient solution due to its ability for mapping based on the cores' runtime load. Besides, through the analyzes of the traffic load, the workload can be distributed between the cores to address network congestion. Based on this, the performance bottleneck can be identified at any core. Nonetheless, owing to its related computational complexity (overhead), implementation of the real-time mapping algorithm incurs not only execution delay but also consumes more energy [24, 25].

In static mapping, during the design time, the application task mapping is performed in the off-line. In this context, the mapping is usually finalized prior to the application execution. Since the related application scenarios are known in the design period, optimal or at least near-optimal solutions can be formulated. This makes the static mapping algorithm a good solution for attending to the associated additional communication overhead of the dynamic mapping. Consequently, the related delay and energy consumption can be minimized. However, static mapping can not handle dynamic scenarios that are usually encountered in nature. Furthermore, static mapping is mainly grouped into exact (mathematical based) and search-based algorithms. Search-based mapping algorithms can be further categorized as heuristic and deterministic (systematic) [24, 25].

In addition, hybrid application mapping algorithms have been presented to address the challenges of the aforementioned mapping algorithms by exploiting their advantages. In this regard, hybrid algorithms offer efficient application mapping solutions. Further information on the NoC mapping algorithmsÂ´ classification can be found in [24, 25]. Besides, another promising area is the multiple layer processing core integration into a 3D design. This can considerably help in reducing the power, area, and delay in signal transmission. Based on this, 3D multicore architectures have been considered as a potential solution for future high-performance systems. However, the related high integration density of the 3D presents additional concern regarding the temperature increase. This effect can bring about high-temperature gradients and thermal hot spots that can make the system unreliable and consequently degraded performance. Therefore, the 3D thermal management demands further research attention [24].

5. Topology performance assessment

Some features determine the performance of the NoC-based system and influence the effectiveness of the related topology implementation. This section presents various topology performance assessment and metrics.

5.1 Topology parameters

Various factors such as bisection width, diameter, degree, and link complexity are some of the parameters that distinguish and characterize one topology from the others. Some of these parameters are discussed in the following subsections.

5.1.1 Node degree

As aforementioned, a network can be regular or irregular if the entire node exhibits the same degree or not. The node degree is the number of edges connected to the node. Moreover, the node degree defines the node's I/O complexity, and depending on the topology, it can vary or constant with the network size. Also, topological features such as constant node degree and smaller degree are normally desirable for a more scalable network. For instance, the required effort in adding new nodes to the existing network is eased by the former feature while the latter facilitates less hardware cost on links. Besides, a constraint always exists on the node degree regarding the number of a node's direct neighbors. Furthermore, it is noteworthy that the node constraint increases due to the communication protocol and hardware limitations. These relate to node degree and port numbers that a node can support. Other factors such as network scalability and space complexity are performance considerations that limit effective node communication.

5.1.2 Diameter

In network topology, the diameter is the maximum shortest distance (path) between the node pairs. Also, in a situation where no direct connection exists between two nodes, the message from the source has to transfers through a number of intermediate nodes to get to its destination. Based on this, multiple hops delay is introduced. Moreover, this delay corresponds to the total number of hops to the destination. Consequently, in network topology, the maximum shortest path length is an important metric. In general, apart from its capability of providing predictable traffic flow and routing paths, a small diameter helps in offering low latency and facilitates network troubleshooting.

5.1.3 Link complexity

In a topology, link complexity defines the aggregate number of links or interconnects. It should be noted that the link complexity is proportional to the network scale and the highest complexity is presented by fully connected networks. Furthermore, when extra links are added to certain networks, their diameters reduce. This can help in offering better communication with lower latency between nodes. However, apart from the introduced complexity, additional links are expensive. Besides, high overhead (i.e. cost, area, etc.) and hardware complexity can also be induced by a high link complexity.

5.1.4 Bisection width

The bisection width is the minimum number of required edges that should be removed so as to divide a network topology into two halves (sub-network) with virtually equal size. It should be noted that a large bisection width is usually desirable for better network stability. This is due to the offered more paths between two sub-network entities and consequently helps in enhancing the overall performance. Also, a large bisection bandwidth can be achieved with a large bisection width. Bisection bandwidth, B_b, can be expressed as

$$B_b = W_b \times B_c \tag{1}$$

where W_b denotes the bisection width and B_c represents the communication channel's bandwidth.

5.2 Performance metrics

There are a number of metrics/parameters that can be employed for assessing the NoC's performance. Some of the performance metrics are presented in this section.

5.2.1 Max end-to-end latency

Latency is the time taken for delivering the packet from the source to the destination. Also, the maximum latency experienced by a given pair of source-destination nodes located at the farthest distance in the network is known as the *Max End-to-End latency* [4, 21], while the metric used to describe the lower latency bound, when there is no other network traffic is the *zero-load latency* [8]. Consider the wormhole switching based NoC, the zero-load latency can be expressed as [26].

$$T_z = \underbrace{\Gamma \cdot t_r}_{\text{Routing delay}} + t_c + \underbrace{L_p/B_c}_{\text{Serialization delay}} \tag{2}$$

where $t_r = t_a + t_s$ represents the router delay, with t_a being the arbitration logic delay and t_s, the switch delay, Γ denotes the average number of routers traversed by a packet to the destination node, t_c, denotes the propagation delay due to communication channel, and L_p represents the length of the packet in bits.

The average latency (delay) can be determined by taking the end-to-end delay mean of each successfully transmitted packet. In the computation of NoC performance, the average network latency is normally employed and can be expressed as [27].

$$T_{av} = \frac{\sum_{i=1}^{p} L_i}{P} \tag{3}$$

where P denotes the number of transmitted flits[8], L_i represents the network latency of the flit i.

It is noteworthy that in the estimation of average end-to-end latency, the packets lost during transmission are not taken into consideration. Also, how swift the packets can be delivered to their destinations indicates the average end-to-end latency, and the larger the value, the less efficient the network [21].

5.2.2 Dropping probability

When packets traverse in network topology, there may be packet loss due to network overloading and transmission errors. The packet loss can be determined by estimating the difference between the total sent packets by the source nodes and those received by the destination nodes. Similarly, the ratio of the dropped (lost) packets to the total sent packets by the source nodes is the dropping probability of the topology. The packet loss, P_l, and the dropping probability, D_p, can be expressed, respectively as [21].

$$P_l = \sum P_g - \sum P_r, \tag{4}$$

$$D_p = \frac{\sum P_d}{\sum P_g}, \tag{5}$$

where P_g denotes the generated packets by the source, P_r represents the packets received by the destination, and P_d is the dropped packets.

[8] The flits are fundamental packets for the execution of link flow control operations and synchronization between routers.

Moreover, owing to the QoS, a low dropping probability rate is preferred. For instance, 0 dropping probability implies no packet drop in the topology, while a 100 dropping probability denotes that the entire packets are dropped [4]. Also, there exist maximum acceptable loss rates for different applications.

5.2.3 Throughput

Throughput, χ, is the rate of packets that are delivered successfully to the destination nodes [4] and can be expressed as [27].

$$\chi = \frac{\sum_{i=1}^{S} P_i}{\tau\zeta} \tag{6}$$

where ζ represents the number of routing nodes, τ denotes the total execution time (total time taken), P_i represents the number of flits that information i contains in time τ, and s is the number of information sent or received in time τ.

Furthermore, the average throughput can be estimated by averaging the number of successfully received packets per second during transmission [21]. Also, when the traffic rate in the NoC is high, the traversing packets in the network will be contending, leading to transmission latency. Then, there will be an injection rate at which the latency will be prohibitively high. This instant is known as the *saturation throughput point* [8].

It is noteworthy that the throughput is mainly contingent on several parameters such as the flow control, employed routing algorithm, available signal-to-noise ratio, available bandwidth, data loss, hardware utilization, buffering, and employed protocol [8]. Also, throughput is based on link utilization in the network. This parameter signifies the number of supported flits by each link in unit time and can be defined as [27].

$$U_\ell = \sum_{i=1}^{s} P_i \tau \frac{\Gamma_{min}}{\Psi} \tag{7}$$

where Γ_{min} is the minimum number of routers traversed by data i and Ψ represents the number of links.

5.2.4 Average queue occupancy

The mean queue length measured as regards packets is the average queue length. Moreover, it can be used to indicates buffer utilization. Therefore, a shorter queue signifies a lower buffer utilization as well as shorter queuing delay. Also, to get the utilization ratio, the queue length is sampled at every time slot [21]. Furthermore, different active queue management techniques such as random early detection, deficit round-robin, fair queuing, drop-tail, stochastic fair queue, and random exponential marking can be employed for packet flow control between various source nodes and destination nodes. This can be achieved through the management of the intermediate routers' buffers [28].

6. NoC challenges

As aforementioned, NoC offers a scalable and modular platform for offering efficient on-chip communication for addressing the trend of SoC integration, however, certain related challenges still demand attention to further enhance the system performance. In this section, we discuss a number of on-chip challenges.

6.1 Links

The choice of parallel or serial links for the data transfer has been one of the primary issues in the NoC due to their associated features. For instance, serial links can considerably save the area, alleviate noise, and reduce interference. However, serializer and de-serializer circuits are required for data transport. On the other hand, the parallel link helps in reducing the power dissipation nevertheless, it consumes more area owing to its buffer-based architecture [10, 29].

6.2 Router architecture

One of the main factors for the embedded systems is the product cost. Besides, the underlying architecture is essential to be small in size and consequently consume less power. Based on this, the routing protocol design presents a tradeoff between cost and performance. For instance, router design will be complicated by a complex routing protocol. In this context, more area and power will be consumed, making it uneconomical. On the other hand, a simpler routing protocol will be a cost-effective solution however, its performance in traffic routing will not be effective [10, 29].

6.3 NoC area/space optimization

In the NoC architecture, communication takes place through the connected modules through the router network by means of long links. Besides, the various schemes such as link sizing, packet sizing, buffer sizing, flow/congestion control, and switching protocol for different topologies not only demand enormous space for NoC design but also make open benchmarks challenging. Consequently, to enhance system performance, link optimization is very imperative. Although the issue can be addressed with repeater implementation, more chip area will be consumed [10, 29]. Similarly, efficient design tools for space evaluation and implementation that can be seamlessly integrated with the current standard tools are required to facilitate extensive employment of NoC technology. Also, because of the complexity of NoC systems, network simulation time will be prohibitive. Therefore, to optimize the simulation speed, innovative techniques are required. Besides, to ensure appropriate architecture selection for an application, open benchmarks are required for different performance features comparison [12].

6.4 Latency

In NoCs, latency increase happens due to additional delay for data packetization/de-packetization at the NIs. It can also be attributed to the fault tolerance protocol overheads and flow/congestion control delays. Besides, owing to contention and buffering, routing delays can also affect network performance. Consequently, to enhance network performance (i.e. to satisfy the stringent latency constraints), native NoC support, low diameter topologies, and advanced flow control approaches are demanded [12].

6.5 Power consumption leakage

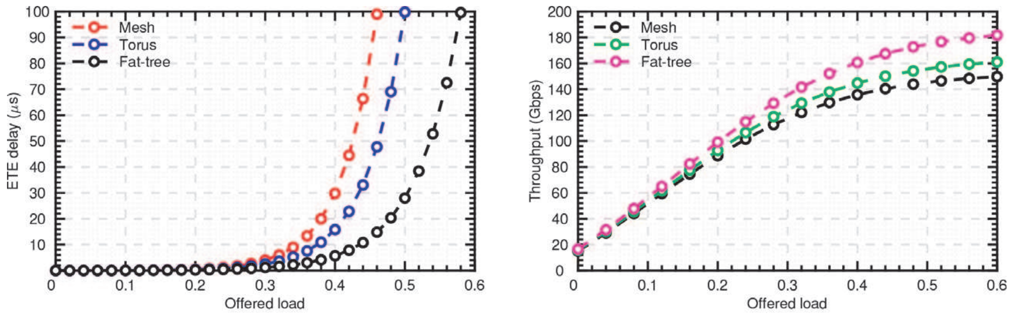
Depending on the application, the link utilization in the NoC may vary and in several cases, it is very low. To meet the worst-case scenario requirements, the NoCs are designed to keep redundant links and function at low link utilization. Nevertheless, even with ideal links, NoC consumes relatively much power owing to the associated complex routing logic blocks and NI. As a result, to further enhance its performance regarding leakage power consumption reduction, innovative architecture and circuit techniques are required [10, 29].

7. Simulation analysis and results

In this section, we consider a 4×4 2D mesh, torus, and fat-tree-based NoCs in the simulation analysis and present results regarding their performance. The simulation is based on the OPNET network simulator. We assumed that there is independent packet generation by the functional cores at time intervals that follow a negative exponential distribution. Also, we assumed a uniform traffic pattern where each processor forwards packets to others with equal probability. Likewise, we use payload packet sizes range from 256 to 1024 bytes for a diverse offered load. End-to-end (ETE) delay (latency) and throughput are the considered performance metrics.

There are general drift patterns in the considered NoCs, resulting in a similar performance exhibition. **Figure 12(a)** depicts the average latency and indicates it increases with an increase in the offered load and rises faster after saturation. For instance, with mesh topology, the average latency is less than 2 μs before the offered load of 0.2 then grows intensely later. This rapid increase after saturation load can be attributed to network congestion. Also, at 70 μs, the offered load for mesh, torus, and fat-tree are about 0.45, 0.48, and 0.56, respectively. Also, **Figure 12(a)** illustrates the throughput with various offered loads and shows it increases with an increase in the offered loads before saturation. For instance, at 0.6 offered load, the throughput of mesh, torus, and fat-tree are about 150, 160, and 180 Gbit/s, respectively.

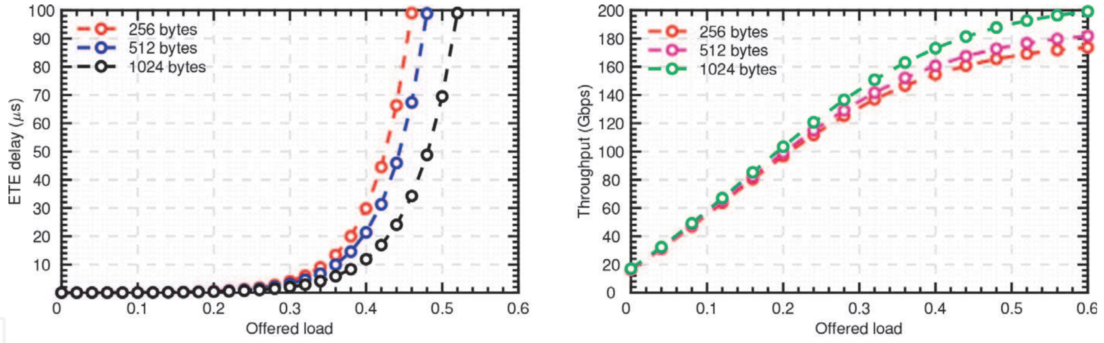
Furthermore, the average latency considering various offered loads and packet lengths is illustrated in **Figure 13(a)**. It is noteworthy that based on the packet sizes, the network saturates at different loads. For instance, the average latency is comparatively low before the saturation load and a considerable surge occurs after it. So, the saturation load for 256 bytes is lower compared with larger packets. Also, with an increase in the packet size, the variation between the average latency curves of



(a) average latency

(b) throughput

Figure 12.
Performance analysis of the considered topologies under different offered loads and at 256 bytes packet.



(a) average latency

(b) throughput

Figure 13.
Performance analysis of 4×4 mesh network under different offered loads and packet lengths.

the adjacent packet sizes turns out to be smaller. Similarly, the network throughput based on different offered loads and packet sizes is illustrated in **Figure 13(b)**.

In general, larger packet length results in lower average latency and higher saturation load. This is owing to the required more transmission time and inter-packet arrival interval by the larger packets compared with smaller ones given the same offered load. Based on this, the path-setup packets blocking possibility can be minimized. Consequently, the destination can effectively receive more packets, resulting in higher throughput and saturation loads.

8. Conclusion

The current and the next-generation applications demand reliable and high-performance on-chip communication for various domain-specific/architecture-aware large-scale multiprocessor system-on-chips/embedded systems. Some of the major research areas in the NoC are topology, routing, and switching. In this chapter, we have presented a comprehensive overview of the evolution of its architectures and have highlighted their associated features. In this context, we have focused mainly on some defining features such as the topology, routing algorithms, and switching arrangements that are promising for the current and future on-chip architectures. Besides, we have presented different application mapping algorithms that are capable of influencing the NoC overall performance, mainly regarding the power requirement and network latency. Also, we have discussed various open problems in its design and implementation. It is noteworthy that the choice of NoC depends mainly on the use cases that will determine the trade-offs between the area, cost, power consumption, and overall performance.

Acknowledgements

This work is supported by the European Regional Development Fund (FEDER), and Internationalization Operational Programme (COMPETE 2020) of the Portugal 2020 (P2020) framework, under the projects DSPMetroNet (POCI-01-0145-FEDER-029405) and UIDB/50008/2020-UIDP/50008/2020 (DigCORE), and by FCT/MCTES through national funds and when applicable co-funded EU funds under the project UIDB/50008/2020-UIDP/50008/2020 (action QuRUNNER).

Author details

Isiaka A. Alimi[1]*, Romil K. Patel[1,2], Oluyomi Aboderin[1], Abdelgader M. Abdalla[1], Ramoni A. Gbadamosi[3], Nelson J. Muga[1], Armando N. Pinto[1,2] and António L. Teixeira[1,2]

1 Instituto de Telecomunicações and University of Aveiro, Portugal

2 Department of Electronics, Telecommunications and Informatics, University of Aveiro, Portugal

3 Faculty of Science, National Open University of Nigeria, Akure, Nigeria

*Address all correspondence to: iaalimi@ua.pt

References

[1] A. Kalita, K. Ray, A. Biswas, and M. A. Hussain. A topology for network-on-chip. In *2016 International Conference on Information Communication and Embedded Systems (ICICES)*, pages 1–7, 2016.

[2] Haytham Elmiligi, Ahmed A. Morgan, M. Watheq El-Kharashi, and Fayez Gebali. Power optimization for application-specific networks-on-chips: A topology-based approach. *Microprocessors and Microsystems*, 33(5):343–355, 2009.

[3] D. Bertozzi and L. Benini. Xpipes: a network-on-chip architecture for gigascale systems-on-chip. *IEEE Circuits and Systems Magazine*, 4(2):18–31, 2004.

[4] T. N. Kamal Reddy, A. K. Swain, J. K. Singh, and K. K. Mahapatra. Performance assessment of different Network-on-Chip topologies. In *2014 2nd International Conference on Devices, Circuits and Systems (ICDCS)*, pages 1–5, 2014.

[5] Zulqar Nain, Rashid Ali, Sheraz Anjum, M. Afzal, and S. Kim. A Network Adaptive Fault-Tolerant Routing Algorithm for Demanding Latency and Throughput Applications of Network-on-a-Chip Designs. *Electronics*, 9(1076):1–18, 2020.

[6] Isiaka A. Alimi, Ana Tavares, Cátia Pinho, Abdelgader M. Abdalla, Paulo P. Monteiro, and António L. Teixeira. *Enabling Optical Wired and Wireless Technologies for 5G and Beyond Networks*, chapter 8, pages 1–31. IntechOpen, London, 2019.

[7] Cátia Pinho, Isiaka Alimi, Mário Lima, Paulo Monteiro, and António Teixeira. *Spatial Light Modulation as a Flexible Platform for Optical Systems*, chapter 7, pages 1–21. IntechOpen, London, 2019.

[8] Haseeb Bokhari and Sri Parameswaran. *Network-on-Chip Design*, chapter 5, pages 461–489. Springer Netherlands, Dordrecht, 2017.

[9] G. Passas, M. Katevenis, and D. Pnevmatikatos. Crossbar NoCs Are Scalable Beyond 100 Nodes. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 31(4): 573–585, 2012.

[10] H. J. Mahanta, A. Biswas, and M. A. Hussain. Networks on Chip: The New Trend of On-Chip Interconnection. In *2014 Fourth International Conference on Communication Systems and Network Technologies*, pages 1050–1053, 2014.

[11] S. Johari and V. Sehgal. Master-based routing algorithm and communication-based cluster topology for 2D NoC. *The Journal of Supercomputing*, 71:4260–4286, 2015.

[12] Sudeep Pasricha and Nikil Dutt. *Networks-On-Chip*, chapter 12, pages 439–471. Systems on Silicon. Morgan Kaufmann, Burlington, 2008.

[13] B. N. K. Reddy, D. Kishan, and B. V. Vani. Performance constrained multi-application network on chip core mapping. *International Journal of Speech Technology*, 22:927–936, 2019.

[14] Evgeny Bolotin and Israel Cidon and Ran Ginosar and Avinoam Kolodny. Cost considerations in network on chip. *Integration*, 38(1):19–42, 2004.

[15] D. Flynn. AMBA: enabling reusable on-chip designs. *IEEE Micro*, 17(4): 20–27, 1997.

[16] Rohita P. Patil and Pratima V. Sangamkar. Review of System-On-Chip Bus Protocols. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, 4(1):271–281, 2015.

[17] María Dolores Valdés Peña Juan José Rodríguez Andina, Eduardo de la Torre Arnanz. *Embedded Processors in FPGA Architectures*.

[18] Mohandeep Sharma and D. Kumar. Wishbone bus architecture - a survey and comparison. *International Journal of VLSI design & Communication Systems (VLSICS)*, 3(2):107–124, 2012.

[19] Altera. Avalon Bus Specification. Reference manual: 2.3, July 2003.

[20] Tobias Bjerregaard and Shankar Mahadevan. A Survey of Research and Practices of Network-on-Chip. *ACM Comput. Surv.*, 38(1):1–es, June 2006.

[21] J. Chen, P. Gillard, and Cheng Li. Network-on-Chip (NoC) Topologies and Performance: A Review. In *Proceedings of the 2011 Newfoundland Electrical and Computer Engineering Conference (NECEC)*, pages 1–6, 2011.

[22] Y. Li and S. Peng. Dual-cubes: A new interconnection network for high-performance computer clusters. In *International Computer Symposium, Workshop on Computer Architecture*, pages 1–7, 2000.

[23] S. Ma, N. E. Jerger, and Z. Wang. DBAR: An efficient routing algorithm to support multiple concurrent applications in networks-on-chip. In *2011 38th Annual International Symposium on Computer Architecture (ISCA)*, pages 413–424, 2011.

[24] W. Amin, F. Hussain, S. Anjum, S. Khan, N. K. Baloch, Z. Nain, and S. W. Kim. Performance Evaluation of Application Mapping Approaches for Network-on-Chip Designs. *IEEE Access*, 8:63607–63631, 2020.

[25] Pradip Kumar Sahu and Santanu Chattopadhyay. A survey on application mapping strategies for Network-on-Chip design. *Journal of Systems Architecture*, 59(1):60–76, 2013.

[26] V. F. Pavlidis and E. G. Friedman. 3-D Topologies for Networks-on-Chip. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 15(10): 1081–1090, 2007.

[27] Xingang Ju and Liang Yang. Performance analysis and comparison of 2×4 network on chip topology. *Microprocessors and Microsystems*, 36(6): 505–509, 2012.

[28] S. Patel and A. Sharma. The low-rate denial of service attack based comparative study of active queue management scheme. In *2017 Tenth International Conference on Contemporary Computing (IC3)*, pages 1–3, 2017.

[29] A. Agarwal and R. Shankar. Survey of Network on Chip (NoC) Architectures & Contributions. volume 3, pages 21–27, 2009.